Database - Feature #1660

add conversion and runtime support for OF keyword use in a CAN-FIND

10/22/2012 12:16 AM - Eric Faulhaber

Status:	Closed	**Start date:**	06/18/2013
Priority:	Normal	**Due date:**	06/26/2013
Assignee:	Ovidiu Maxiniuc	**% Done:**	100%
Category:		**Estimated time:**	40.00 hours
Target version:	Runtime Support for Server Features		
billable:	No	**vendor_id:**	GCD
Description			

History

#1 - 10/22/2012 12:20 AM - Eric Faulhaber

Actually not sure if this currently is supported or not (most likely not). Need some test cases to determine the limits of this support and some customer report results to define the actual, required scenarios.

Note that this could convert different ways, depending on whether the tables referenced in the CAN-FIND reside in the same database or not. If in the same database, we will use a sub-select approach. If in separate databases, we will use a client-side where clause approach (yuck).

#2 - 10/31/2012 04:03 PM - Greg Shah

- Target version set to Milestone 7

#3 - 12/21/2012 12:54 PM - Eric Faulhaber

- Assignee set to Stanislav Lomany

I still need to review customer reports to determine actual scenarios, but please start with some basic use cases.

#4 - 02/21/2013 10:50 PM - Eric Faulhaber

- Assignee changed from Stanislav Lomany to Vadim Nebogatov

The construct record OF table in a 4gl query signifies a join between two tables (see "Record phrase" documentation in ABL Reference for details). We need to know whether we currently support this construct properly within a CAN-FIND function.

Please create some test cases which test conversion support for the following scenarios:

```
CAN-FIND(FIRST record OF table)
CAN-FIND(FIRST record OF table WHERE ...)
CAN-FIND(record OF table)
CAN-FIND(record OF table WHERE ...)
```

Use permanent tables for record and table. There are some tables (i.e., Person, Address, Pers-Addr) appropriate for this type of use in the p2j_test database in the testcases project.

These are the only scenarios we need to support for M4. No need to mess with lock options or temp-table tests.

Please post your results/findings here.

#5 - 02/23/2013 12:38 AM - Vadim Nebogatov

I have created environment for tests and made some first conversion tests. Will try to make more tests on Monday.
What I found so far.

```
CAN-FIND(FIRST address of address where address.addr-id = pers-addr.addr-id)
```

is converted properly, but conversion of

```
CAN-FIND(FIRST address FIELDS (street) of address where address.addr-id = pers-addr.addr-id)
```

gives compilation error.

Test I used wasbased on JoinIndex1.p:

```
for each person no-lock,
    each pers-addr no-lock where pers-addr.site-id = person.site-id
                             and pers-addr.emp-num = person.emp-num,
    each address no-lock where address.addr-id = pers-addr.addr-id:

  display
    CAN-FIND(FIRST address FIELDS (street) of address where address.addr-id = pers-addr.addr-id)
    person.site-id
    person.emp-num
    person.first-name
    person.last-name
    pers-addr.active
    pers-addr.link-type
    address.addr-id
    address.city
    address.state
    .

end.
```

#6 - 02/23/2013 01:42 AM - Eric Faulhaber

Vadim Nebogatov wrote:

> ... but conversion of
>
> ```
> CAN-FIND(FIRST address FIELDS (street) of address where address.addr-id = pers-addr.addr-id)
> ```
>
> gives compilation error.

This doesn't look like legal 4gl syntax. I would not expect it to convert.

I just tried the following:

```
find first person no-lock.

/* case 1:  simple can-find with join */
if can-find(first pers-addr of person) then
  message "Found it!".

/* case 2:  simple can-find with join and where clause */
if can-find(first pers-addr of person where pers-addr.site-id > 0) then
  message "Found it!".

/* case 3:  nested can-find with join */
find first person
     where person.emp-num > 0
       and can-find(first pers-addr of person)
     no-lock.

/* case 4:  nested can-find with join and where clause */
find first person
     where person.emp-num > 0
       and can-find(first pers-addr of person where pers-addr.site-id > 0)
     no-lock.
```

These statements convert to:

```
new FindQuery(person, (String) null, null, "person.siteId asc, person.empNum asc", LockType.NONE).first();
/* case 1:  simple can-find with join */
if ((new FindQuery(persAddr, (String) null, null, "persAddr.siteId asc, persAddr.empNum asc", person, LockType
.NONE).hasAny()).booleanValue())
{
   message("Found it!");
}
/* case 2:  simple can-find with join and where clause */
if ((new FindQuery(persAddr, "persAddr.siteId > 0", null, "persAddr.siteId asc, persAddr.empNum asc", person, 
LockType.NONE).hasAny()).booleanValue())
{
   message("Found it!");
}
/* case 3:  nested can-find with join */
new FindQuery(person, "person.empNum > 0 and", null, "person.siteId asc, person.empNum asc", LockType.NONE).fi
rst();
/* case 4:  nested can-find with join and where clause */
new FindQuery(person, "person.empNum > 0 and exists(from PersAddr as persAddr where persAddr.siteId > 0)", nul
l, "person.siteId asc, person.empNum asc", LockType.NONE).first();
```

Conversion of the standalone CAN-FINDs (cases 1 & 2) look correct to me.

Conversion of the nested CAN-FINDs (cases 3 & 4) is incorrect. The HQL is wrong, as it does not take the join between Person and Pers-Addr into account.

Unfortunately, there is one instance of a CAN-FIND using OF embedded within the WHERE clause of an outer FIND statement, so we need to fix this. It matches case 3 (there is no WHERE clause in the CAN-FIND). The code will compile without this fix, but it will fail at runtime.

The correct HQL for case 3 would be:

```
"person.empNum > 0 and exists(from PersAddr as persAddr where persAddr.siteId = person.siteId and persAddr.emp
Num = person.empNum)"
```

This is the only error relevant to the current project. Investigation of other forms of this function is not needed at this time.

#7 - 02/25/2013 02:45 AM - Eric Faulhaber

- Assignee deleted (Vadim Nebogatov)

#8 - 04/18/2013 11:24 PM - Eric Faulhaber

- Estimated time changed from 16.00 to 32.00

#9 - 04/25/2013 03:46 PM - Eric Faulhaber

- Estimated time changed from 32.00 to 40.00

- Due date set to 06/26/2013

- Assignee set to Eric Faulhaber

- Start date set to 06/18/2013

#10 - 08/01/2013 02:17 PM - Eric Faulhaber

- Assignee changed from Eric Faulhaber to Ovidiu Maxiniuc

#11 - 08/08/2013 06:42 AM - Ovidiu Maxiniuc

My first approach here was to analyze the AST tree to see why the OF node does not get encoded as needed.
I used a combination of ASTs from case 1, 3 and 4 above to see the differences. I have no indication that external WHERE clause in sample 2 is not converted correctly.
I noticed that it has a hql="false" annotation. I have forced it to be processed and I got the following generated code (for case 3 above):

```
new FindQuery(person, "person.empNum > 0 and ?", null, "person.siteId asc, person.empNum asc", new Object[]
{
   new FindQuery(persAddr, "persAddr.siteId > 0", null, "persAddr.siteId asc, persAddr.empNum asc", person, is
GreaterThan(persAddr.getSiteId(), 0), LockType.NONE).hasAny()
}).first();
```

This is obviously wrong.

It turned out that I have to conditionally inject the OF join code into the nested record_phrase hql_where/hql_where_alt annotations when the OF clause is used in a such a node with nested_can_find = "true" annotation.
One more issue here: the hql_where/hql_where_alt were present only if the nested CAN-FIND would have a WHERE clause.
So I had to add this exception when the WHERE clause is absent (case 3) the annotation will contain only the OF join clause. In other cases, current the hql_where/hql_where_alt expressions will be AND-ed with OF constraints.

At this moment, the generated code for cases 3 and 4 looks like:

```
new FindQuery(person, "person.empNum > 0 and exists(from PersAddr as persAddr where persAddr.siteId = person.s
iteId and persAddr.empNum = person.empNum)", null, "person.siteId asc, person.empNum asc", LockType.NONE).firs
t();

new FindQuery(person, "person.empNum > 0 and exists(from PersAddr as persAddr where (persAddr.siteId = person.
siteId and persAddr.empNum = person.empNum) and (persAddr.siteId > 0))", null, "person.siteId asc, person.empN
um asc", LockType.NONE).first();
```

I am working now on correctly building the persAddr.siteId = person.siteId and persAddr.empNum = person.empNum phrase as parts of it are temporarily hardcoded. Probably in the evening I will upload a first version of the patch.

#12 - 08/08/2013 01:21 PM - Ovidiu Maxiniuc

- File om_upd20130808a.zip added

The first version of the update attached.
I used the disjunction between of-join foreign-key phrase and any existing where clauses of the nested can-find using parentheses, the extra analysis of removing them if not needed does not worth the extra CPU ticks.

#13 - 08/08/2013 03:37 PM - Eric Faulhaber

Code review 20130808a:

This update looks OK, but it is in a very sensitive area of the conversion, and it is hard to know whether it is safe from code review alone. Please do a conversion-only regression test, to be sure we haven't broken things. If that turns out OK, check it in.

#14 - 08/09/2013 02:37 PM - Ovidiu Maxiniuc

- File om_upd20130809c.zip added

Indeed, there were some issues at conversion time:
- join of a buffer with a table
- join of a buffer with the same table (CAN-FIND buffOfTable1 OF Table1 - I don't know id this is even correct)
- nested CAN-FIND without OF was broken in some particular cases.

I noticed that some processing is done multiple times (namely the SchemaWorker.recordJoin() if method is called with same arguments). I did not investigate further, but isn't it possible to use the cached TableRelation from joins map ?

These issues were fixed and I am optimistic about the re-run of the conversion test I started now.

#15 - 08/09/2013 03:18 PM - Eric Faulhaber

Ovidiu Maxiniuc wrote:

> Indeed, there were some issues at conversion time:
> - join of a buffer with a table
> - join of a buffer with the same table (CAN-FIND buffOfTable1 OF Table1 - I don't know id this is even correct)

If it works in Progress, then it is a correct case, though I don't think we have any like this in any real project

> - nested CAN-FIND without OF was broken in some particular cases.

I guess these last 2 did not come out of regression testing; I didn't think we had any cases like that in the code used for regression testing.

> I noticed that some processing is done multiple times (namely the SchemaWorker.recordJoin() if method is called with same arguments). I did not investigate further, but isn't it possible to use the cached TableRelation from joins map ?

I think it should be OK.

> These issues were fixed and I am optimistic about the re-run of the conversion test I started now.

Great!

#16 - 08/12/2013 12:31 PM - Ovidiu Maxiniuc

- File om_upd20130812b.zip added

There were some nodes duplicated in the /src/aero/timco/majic/dmo/dmo_index.xml. After investigations I had to change the lookup key because in Progress joins between tables are somewhat "reflexive". This means
FIND FIRST Pers-addr OF Person.
and

FIND FIRST Person OF Pers-addr.
will share the same relation, the statements above are equivalent to:
FIND FIRST Pers-addr WHERE Person.site-id = Pers-addr.site-id AND Person.emp-num Pers-addr.emp-num.
and respectively,
FIND FIRST Person WHERE Person.site-id = Pers-addr.site-id AND Person.emp-num = Pers-addr.emp-num.

The new conversion test was successful. There is, in fact, just one file altered, the /src/aero/timco/majic/dmo/dmo_index.xml, but the differences are caused just by the order of the foreign nodes within the class. This is because now the relations between the tables are stores in a Hashtable instead of a HashSet.

#17 - 08/12/2013 12:47 PM - Eric Faulhaber

20130812b update looks good. Please check it in.

#18 - 08/12/2013 02:32 PM - Ovidiu Maxiniuc

Checked in bzr as rev 10375 and distributed by mail.

LE: I forgot to mention that I re-run the conversion test since the bzr repository has changes meanwhile. Only expected changes occurred.

#19 - 08/12/2013 10:21 PM - Eric Faulhaber

- % Done changed from 0 to 100

- Status changed from New to Closed

#20 - 11/16/2016 11:43 AM - Greg Shah

- Target version changed from Milestone 7 to Runtime Support for Server Features

Files

om_upd20130808a.zip	27.8 KB	08/08/2013	Ovidiu Maxiniuc
om_upd20130809c.zip	28.1 KB	08/09/2013	Ovidiu Maxiniuc
om_upd20130812b.zip	28.5 KB	08/12/2013	Ovidiu Maxiniuc